

CB

## BEST AVAILABLE COPY

700 IBM Technical Disclosure Bulletin  
Armonk, NY, US

Vol. 36 No. 1 January 1993

G06FAS140B1F



XP 000333761

OS/2 EE DATABASE MANAGER SQLJRA REMOTE PROTOCOL

033 - 36

Disclosed is a design of Remote Data Service Component of OS/2\* Database Manager. Remote Data Service is a set of functions for OS/2 Extended Edition (EE) that allowed users to access databases not only on their local workstations but also on remote workstations. A private communication protocol (DRDA-0) was implemented in 1.3 EE to access relational database remotely. DRDA-0 is based on Distributed Relational Database Architecture (DRDA), which is the architecture for connecting database products. DRDA basically makes use of an intermediate language defined in Distributed Data Management (DDM) architecture to pass information back and forth from client to server.

Performance results showed that DRDA-0 performed poorly in a homogeneous environment (OS/2 EE client to OS/2 EE server). The main reason for the poor performance is because of the translation overhead from the internal data structure to DDM commands and vice versa.

The design of SQLJRA remote protocol takes advantage of the fact that the data structures built on either the client or the server in a homogeneous environment are the same. Therefore, the underlying data structures are passed directly, without the conversion step being necessary. In summary, the SQLJRA remote protocol has the following advantages:

- (1) Much simpler design to achieve optimum performance. Data is passed in its native form. Performance results showed that the SQLJRA client and server ran four times faster than DRDA-0 client and server. It also achieves up to 56 percent speed-up in throughput and 36 percent speed-up in end-user response time.
- (2) Smaller client and server. Because of the simple design, developers can implement full Remote Data Services functions with only one fourth of the DRDA-0 code (3000 C statements vs. 12,000 C statements). Thus, it helps reduce the RAM and disk space required for both the client and the server.
- (3) Easy vendor connection. This protocol makes use of the published SQLJRA interface to connect clients and servers. The vendor can connect to database servers or clients easily through this published gateway interface.
- (4) Portable design and implementation. The SQLJRA control block structure used is environment independent. The design also uses the OSS (Operating System Services) provided in ES 1.0 Database Manager.

BEST AVAILABLE COPY

OSS provides a platform-independent operating system interface for the OS/2 Database Manager common code development. The SQLJRA client and server can be easily ported to other platforms (e.g., AIX\*) with OSS support.

The precompiler in the OS/2 ES 1.0 Database Manager takes source language SQL statements and maps those statements into Database Manager application programming interfaces (APIs). For each SQL statement there are one or more procedure calls which create a series of data structures for the database call. These data structures are the SQLRA (SQL Request Area) and the SQLDA (SQL Data Area). The SQLRA is then mapped to SQLJRA before being routed to the remote machine. The gateway router will route SQLJRA and SQLDA to host database engine through DRDA-1 protocols or to another OS/2 Database Manager through the SQLJRA remote protocol. The design for the SQLJRA remote protocol takes advantage of the fact that the data structures built on either the client or the server are the actual data structures needed by the application on the other side. Therefore, the underlying data structures are passed directly, without the conversion being necessary. The design has a data stream construction routine on the client and a data structure reconstruction routine on the server to pass the data structures from client to server.

The remote database request follows the following steps:

- (1) The Database Manager run-time APIs and gateway router create engine data structures SQLJRA and SQLDA.
- (2) The gateway router will route the request to the appropriate remote server machine.
- (3) The data stream construction routine will put data in the data structure into a data stream. If the data structure contains pointers, the actual data pointed to by those pointers will be put into the data stream.
- (4) The transport layer communication protocol (APPN or NETBIOS) will transfer the data string from the client to the server.
- (5) The data structure reconstruction routine on the server will convert the byte string back to SQLJRA and SQLDA. The pointers in the data structures will be updated to point to the actual data.
- (6) SQLJRA and SQLDA will be passed to the server database engine. If the server machine contains the target database, a database procedure will be invoked to process this SQL request. If not, the SQLJRA and SQLDA will be forwarded to the next destination through DRDA-1 or SQLJRA remote protocol.

BEST AVAILABLE COPY

# BEST AVAILABLE COPY

OS/2 EE DATABASE MANAGER SQLJRA REMOTE PROTOCOL - Continued

(7) The engine will put the result into output SQLDA and SQLCA (SQL communication area). The data stream construction routine on the server will put data in output SQLDA and SQLCA into a byte string.

(8) The transport layer communication protocol will transfer the output data back to the client.

(9) The data structure reconstruction routine on the requester will get the output data from the data stream and put them into the user output SQLDA.

The following figure shows the performance result of TPCA (Transaction Processing Council A) benchmark, which is an industry standard relational database performance benchmark.

Throughput: (# of transactions per second)

	DRDA-0	JRA	Improvement
1 user :	1.353	1.838	40%
10 users:	5.429	7.826	44%
20 users:	4.705	7.340	56%
40 users:	4.033	6.018	49%

Response time: (seconds)

	DRDA-0	JRA	Improvement
1 user :	0.739	0.544	26%
10 users:	1.842	1.278	31%
20 users:	4.251	2.725	36%
40 users:	9.912	6.645	33%

This article describes an efficient design and implementation of remote relational database access. The SQLJRA remote protocol design is unique because:

- (1) It is an OS/2 Database Manager-specific application layer protocol.
- (2) It uses the published SQLJRA interface to facilitate easy vendor connection.
- (3) Portable design and implementation. It utilizes the SQLJRA gateway interface which is extendable and environment independent. It also uses the OSS provided in ES 1.0 Database Manager. The same implementation can be easily ported to an AIX platform.

The SQLJRA remote protocol implementation has the following advantages over the previous designs:

OS/2 EE DATABASE MANAGER SQLJRA REMOTE PROTOCOL - Continued

(1) Substantially improved performance. The SQLJRA remote protocol achieves up to 56 percent speed-up in throughput and 36 percent speed-up in end-user response time compared with the remote database service implementation in OS/2 EE 1.3.

(2) Skinny client and server. The SQLJRA remote protocol implementation is also much smaller (3000 C statements vs. 12,000 C statements), thus helping to reduce the RAM and disk space required for the client and the server.

The disclosed SQLJRA remote protocol is in OS/2 Extended Services 1.0.

\* Trademark of IBM Corp.

**BEST AVAILABLE COPY**